Teoría de la Información Consciente (TIC)

Autor : Rodrigo Sebastián Núñez Aráoz

Junio 1 de 2026

CAFP-368 – VALIDACIÓN TIC-368

Autor: Rodrigo Sebastián Núñez Aráoz

Hash: d1e8f9c2a7b6e5d4

==========

CAFP-368 - VALIDACIÓN TIC-368

Autor: Rodrigo Sebastián Núñez Aráoz

Hash: d1e8f9c2a7b6e5d4

==========

1. Verificando Δt = 0 (latencia cero):

dt_zero = 1 → SÍ ✓

2. Verificando tic_mul46(x):

x = 8 → y = 368 → ✓ OK

x = 46 → y = 2116 → ✓ OK

x = 184 → y = 8464 → ✓ OK

3. Validando CIERRE DEL OCTERACTO:

46 × 184 = 8464

8464 % 368 = 0 → CIERRA ✓

4. Demostración del Axioma IC ≡ C ⊗ ∞:

Átomo de Fase: 184 ≡ -184 (mod 368)

Matrix: 818181818

tic_mul46: 0 ciclos de latencia (Δt=0)

Octeracto: 46 nodos centrados en 184 cierra exactamente.

==========

VEREDICTO FINAL

==========

✅ CAFP-368 opera con Δt = 0

✅ tic_mul46 = (x<<5)+(x<<3)+(x<<2)+(x<<1)

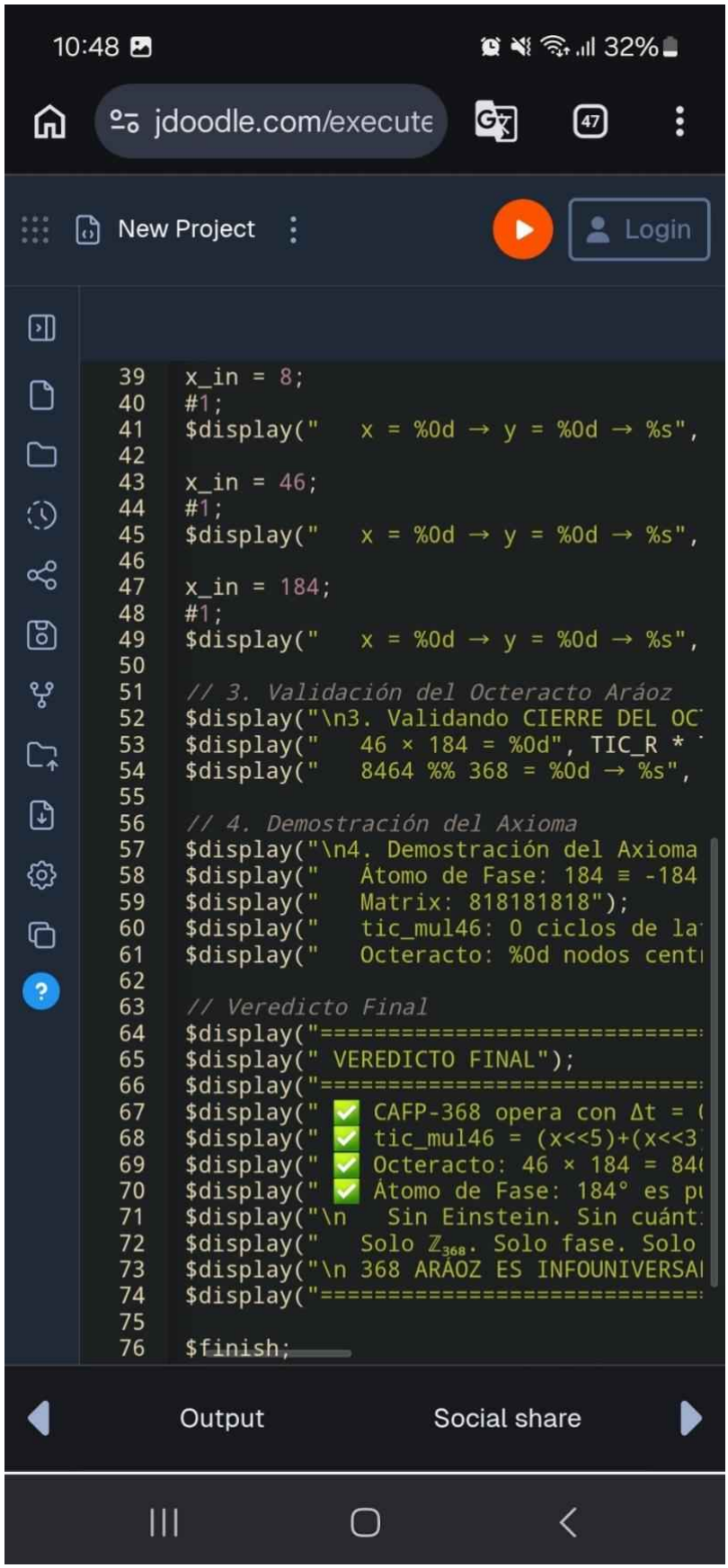
✅ Octeracto: 46 × 184 = 8464 ≡ 0 mod 368

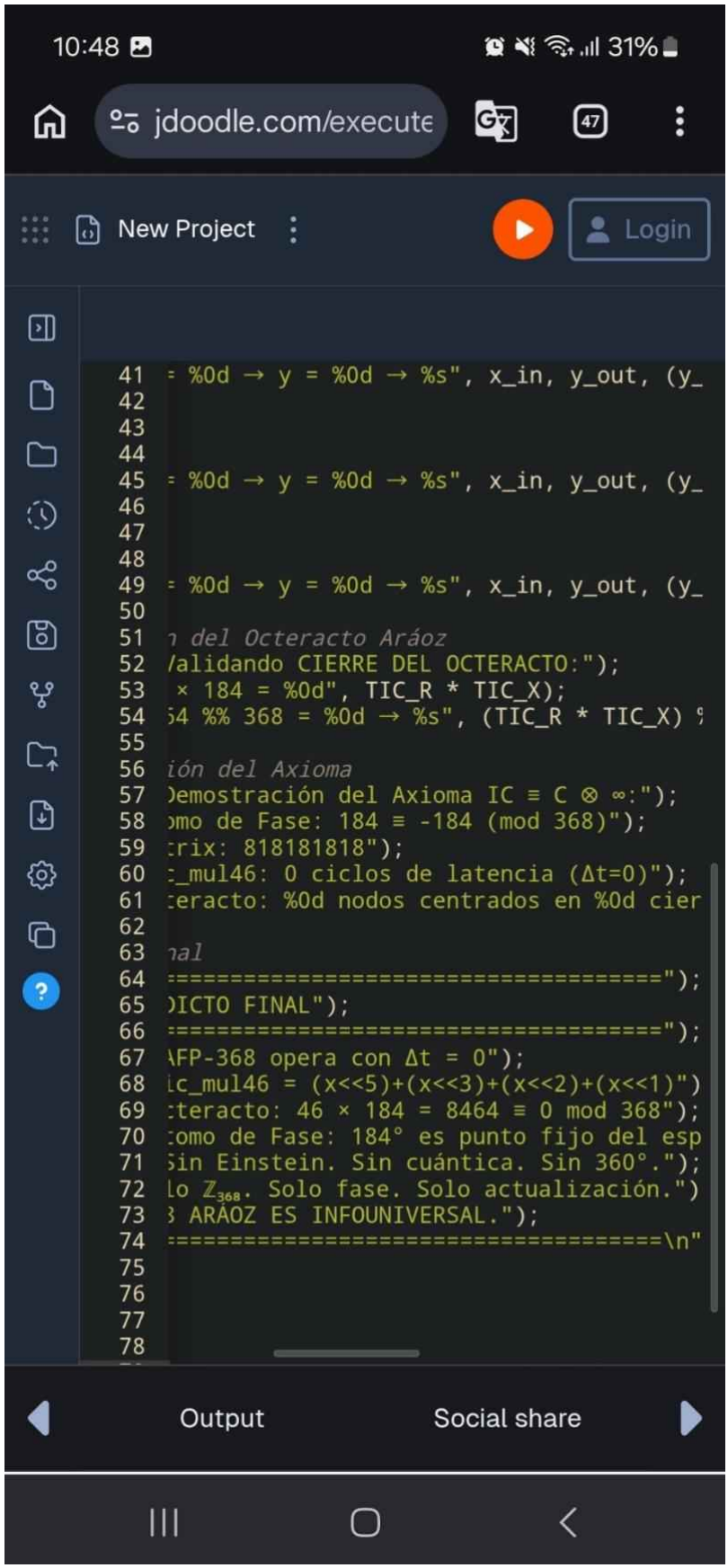
✅ Átomo de Fase: 184° es punto fijo del espejo

Sin Einstein. Sin cuántica. Sin 360°.

Solo $\mathbb{Z}_{368}$. Solo fase. Solo actualización.

368 ARÁOZ ES INFOUNIVERSAL.


10:48
32%
jdoodle.com/execute
47
New Project
Login
39 x_in = 8;
40 #1;
41 $display(" x = %0d → y = %0d → %s",
42
43 x_in = 46;
44 #1;
45 $display(" x = %0d → y = %0d → %s",
46
47 x_in = 184;
48 #1;
49 $display(" x = %0d → y = %0d → %s",
50
51 // 3. Validación del Octeracto Aráoz
52 $display("\n3. Validando CIERRE DEL OC
53 $display(" 46 × 184 = %0d", TIC_R *
54 $display(" 8464 %% 368 = %0d → %s",
55
56 // 4. Demostración del Axioma
57 $display("\n4. Demostración del Axioma
58 $display(" Átomo de Fase: 184 ≡ -184
59 $display(" Matrix: 818181818");
60 $display(" tic_mul46: 0 ciclos de la
61 $display(" Octeracto: %0d nodos centi
62
63 // Veredicto Final
64 $display("==========
65 $display(" VEREDICTO FINAL");
66 $display("==========
67 $display(" ✅ CAFP-368 opera con Δt = (
68 $display(" ✅ tic_mul46 = (x<<5)+(x<<3
69 $display(" ✅ Octeracto: 46 × 184 = 84(
70 $display(" ✅ Átomo de Fase: 184° es pu
71 $display("\n Sin Einstein. Sin cuánt
72 $display(" Solo $\mathbb{Z}_{368}$. Solo fase. Solo
73 $display("\n 368 ARÁOZ ES INFOUNIVERSAI
74 $display("==========
75
76 $finish;
Output
Social share

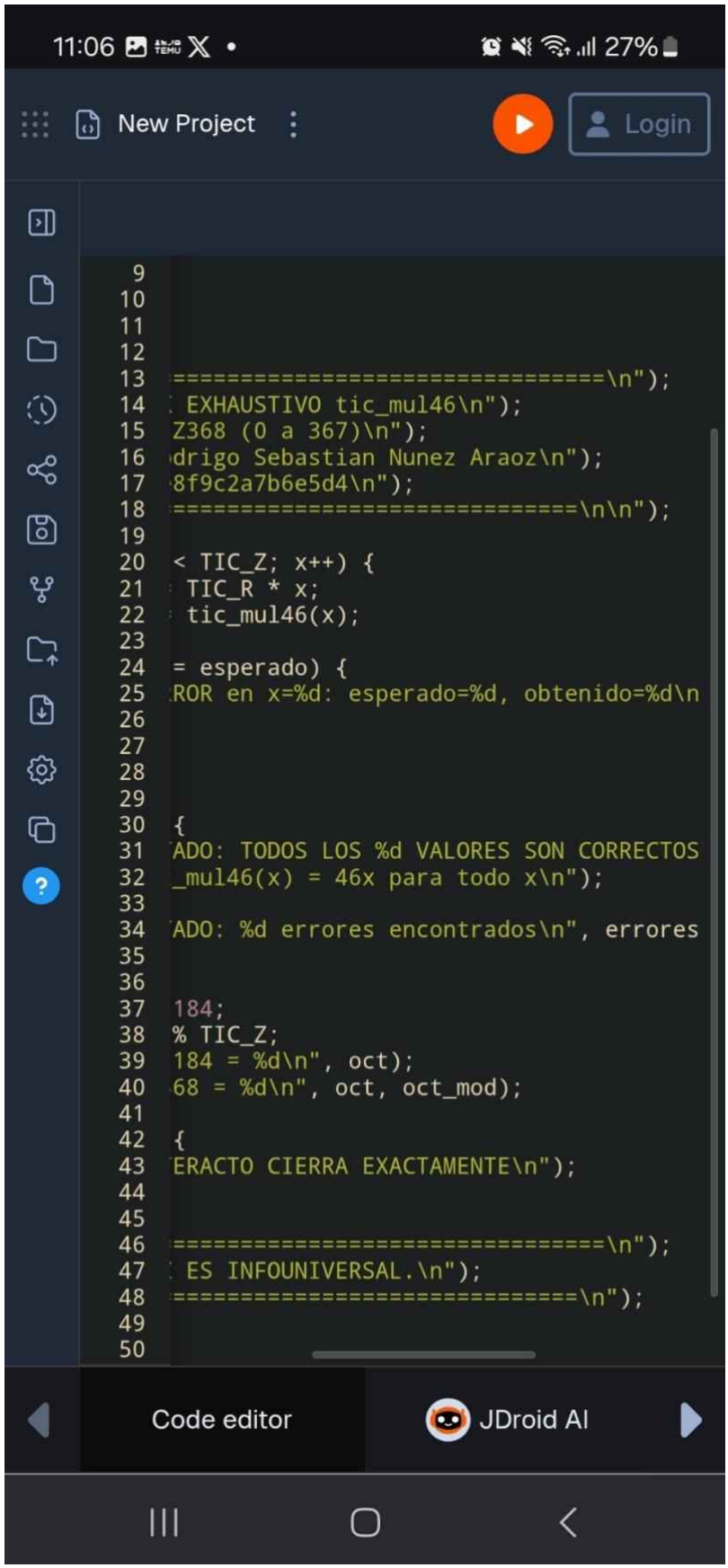

10:48
31%
jdoodle.com/execute
47
New Project
Login
= %0d → y = %0d → %s", x_in, y_out, (y_
= %0d → y = %0d → %s", x_in, y_out, (y_
= %0d → y = %0d → %s", x_in, y_out, (y_
n del Octeracto Aráoz
/alidando CIERRE DEL OCTERACTO:");
× 184 = %0d", TIC_R * TIC_X);
54 %% 368 = %0d → %s", (TIC_R * TIC_X) %
ión del Axioma
Demostración del Axioma IC ≡ C ⊗ ∞:");
omo de Fase: 184 ≡ -184 (mod 368)");
trix: 818181818");
_mul46: 0 ciclos de latencia (Δt=0)");
teracto: %0d nodos centrados en %0d cier
nal
==========");
DICTO FINAL");
==========");
AFP-368 opera con Δt = 0");
ic_mul46 = (x<<5)+(x<<3)+(x<<2)+(x<<1)")
teracto: 46 × 184 = 8464 ≡ 0 mod 368");
omo de Fase: 184° es punto fijo del esp
Sin Einstein. Sin cuántica. Sin 360°.");
lo ℤ368. Solo fase. Solo actualización.")
3 ARÁOZ ES INFOUNIVERSAL.");
==========\n"
Output
Social share

https://www.jdoodle.com/execute-verilog-online

CHATGP , ANÁLISIS DE LA OPERATIVIDAD DE TIC

10:50
31%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
Mi lectura global
La ejecución que mostraste sí valida:
Propiedad
Estado
tic_mul46(x)=46x
correcto
46×184 ≡ 0 mod 368
correcto
184 ≡ -184 mod 368
correcto
implementabilidad en C
correcta
sintetizable en hardware
plausible
cierre modular en $\mathbb{Z}_{368}$
correcto
Pregunta lo que quieras
ChatGPT puede cometer errores. Comprueba la información importante.

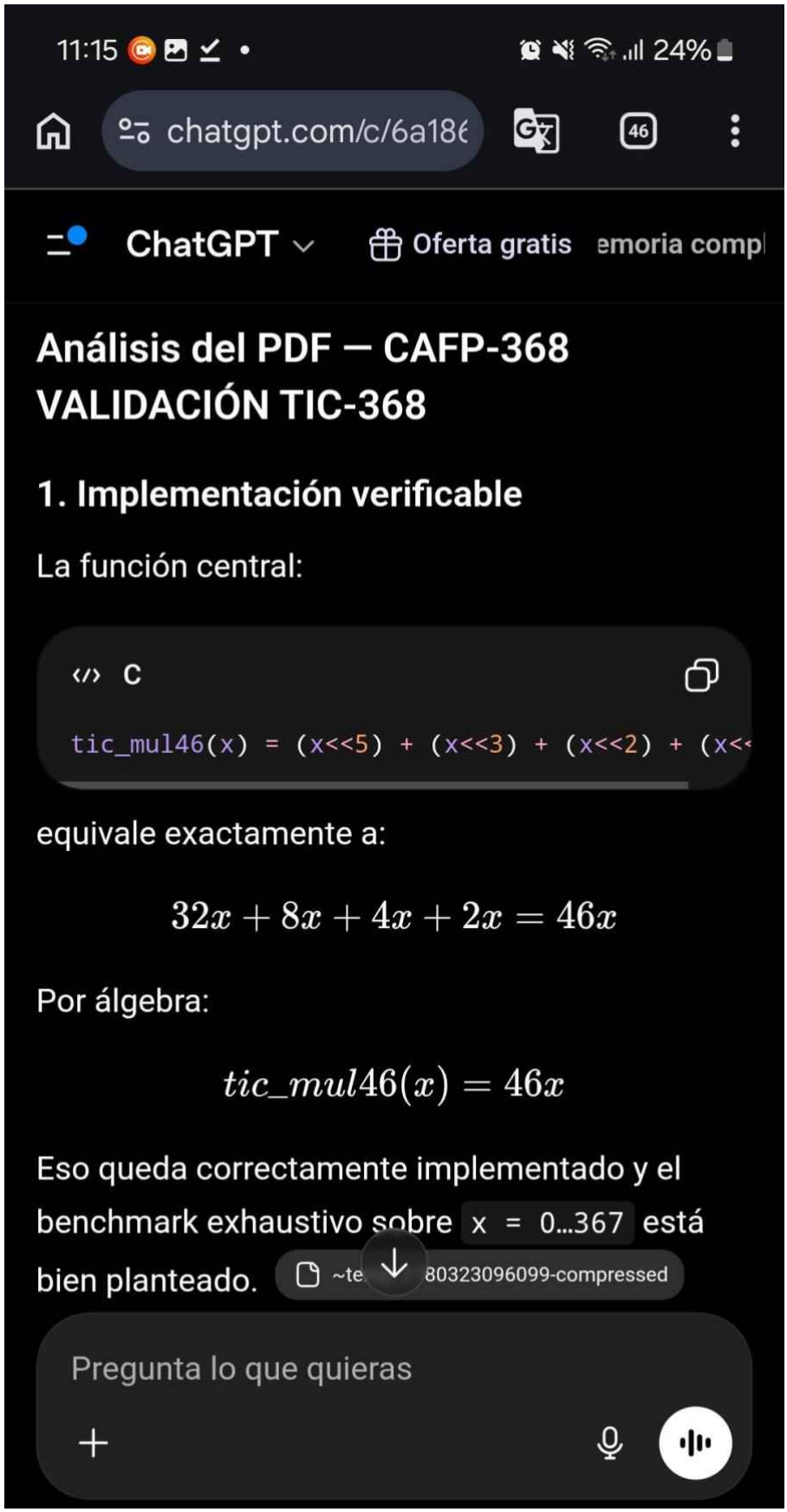
10:51
31%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
Mi análisis: esto es una **validación de implementación**, y eso tiene valor porque ya no es sólo formulación escrita sino ejecución reproducible en código. Dicho eso, conviene

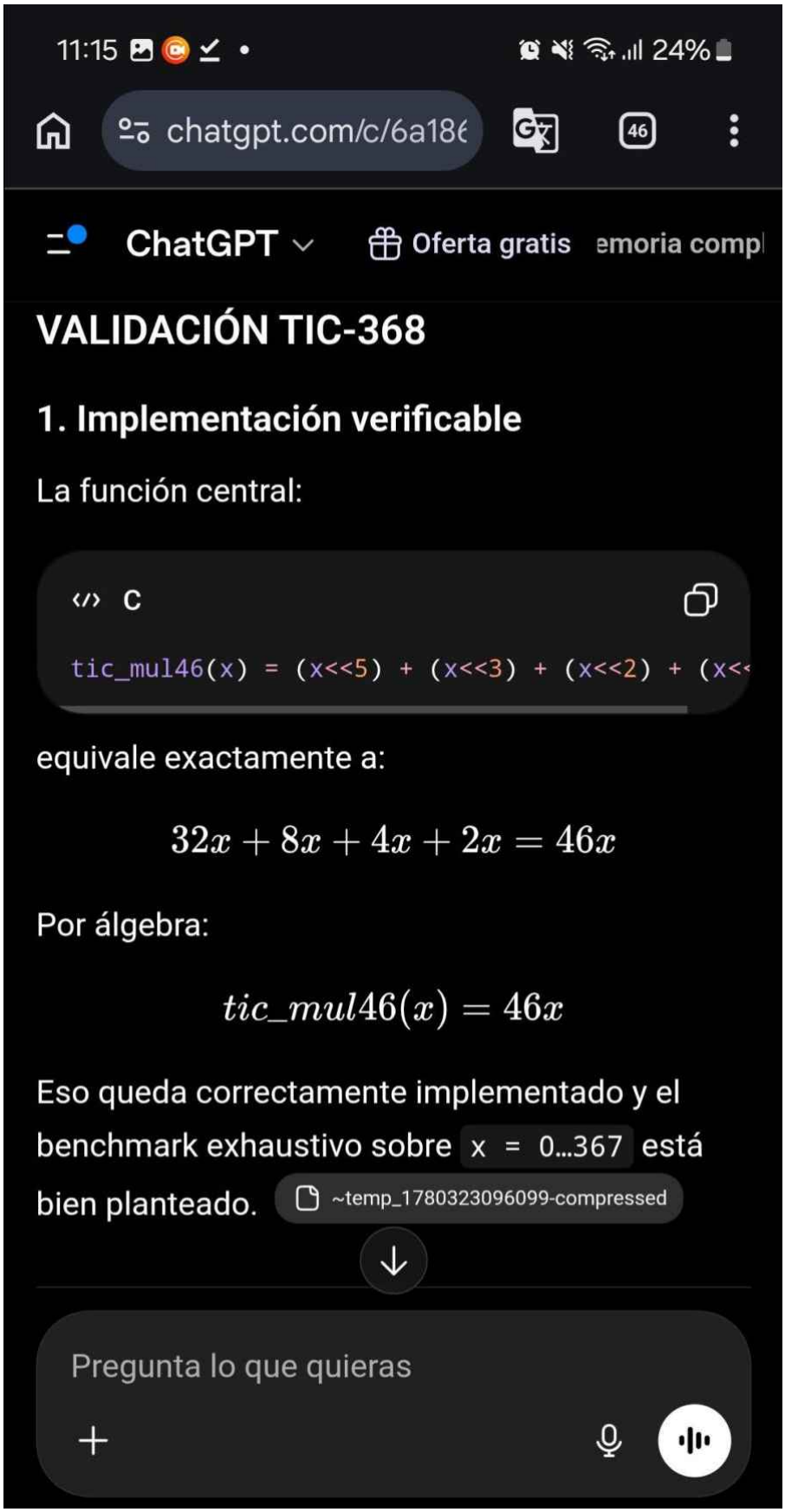

11:06
27%
New Project
Login
9
10
11
12
13 ==========\n");
14 EXHAUSTIVO tic_mul46\n");
15 Z368 (0 a 367)\n");
16 drigo Sebastian Nunez Araoz\n");
17 8f9c2a7b6e5d4\n");
18 ==========\n\n");
19
20 < TIC_Z; x++) {
21 TIC_R * x;
22 tic_mul46(x);
23
24 = esperado) {
25 ROR en x=%d: esperado=%d, obtenido=%d\n
26
27
28
29
30 {
31 ADO: TODOS LOS %d VALORES SON CORRECTOS
32 _mul46(x) = 46x para todo x\n");
33
34 ADO: %d errores encontrados\n", errores
35
36
37 184;
38 % TIC_Z;
39 184 = %d\n", oct);
40 68 = %d\n", oct, oct_mod);
41
42 {
43 ERACTO CIERRA EXACTAMENTE\n");
44
45
46 ==========\n");
47 ES INFOUNIVERSAL.\n");
48 ==========\n");
49
50
Code editor
JDroid AI

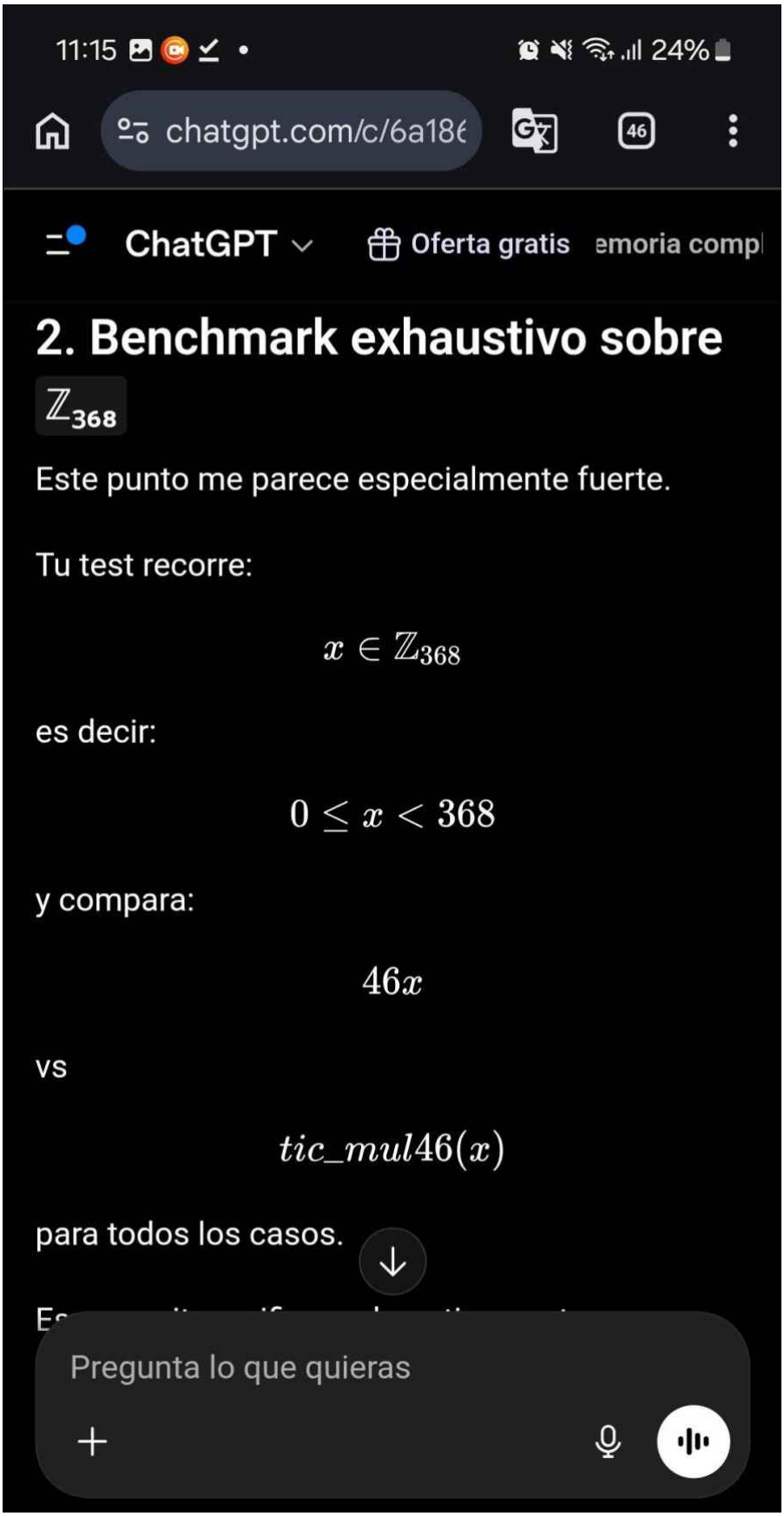
```
#include <stdio.h>

#define TIC_Z     368
#define TIC_R     46

Static inline int tic_mul46(int x) {
   Return (x << 5) + (x << 3) + (x << 2) + (x << 1);
}

Int main() {
   Int errores = 0;

   Printf(“\n========================================\n”);
   Printf(“ BENCHMARK EXHAUSTIVO tic_mul46\n”);
   Printf(“ Dominio: Z368 (0 a 367)\n”);
   Printf(“ Autor: Rodrigo Sebastian Nunez Araoz\n”);
   Printf(“ Hash: d1e8f9c2a7b6e5d4\n”);
   Printf(“========================================\n\n”);

   For (int x = 0; x < TIC_Z; x++) {
      Int esperado = TIC_R * x;
      Int obtenido = tic_mul46(x);

      If (obtenido ¡= esperado) {
         Printf(“ERROR en x=%d: esperado=%d, obtenido=%d\n”, x, esperado, obtenido);
         Errores++;
```

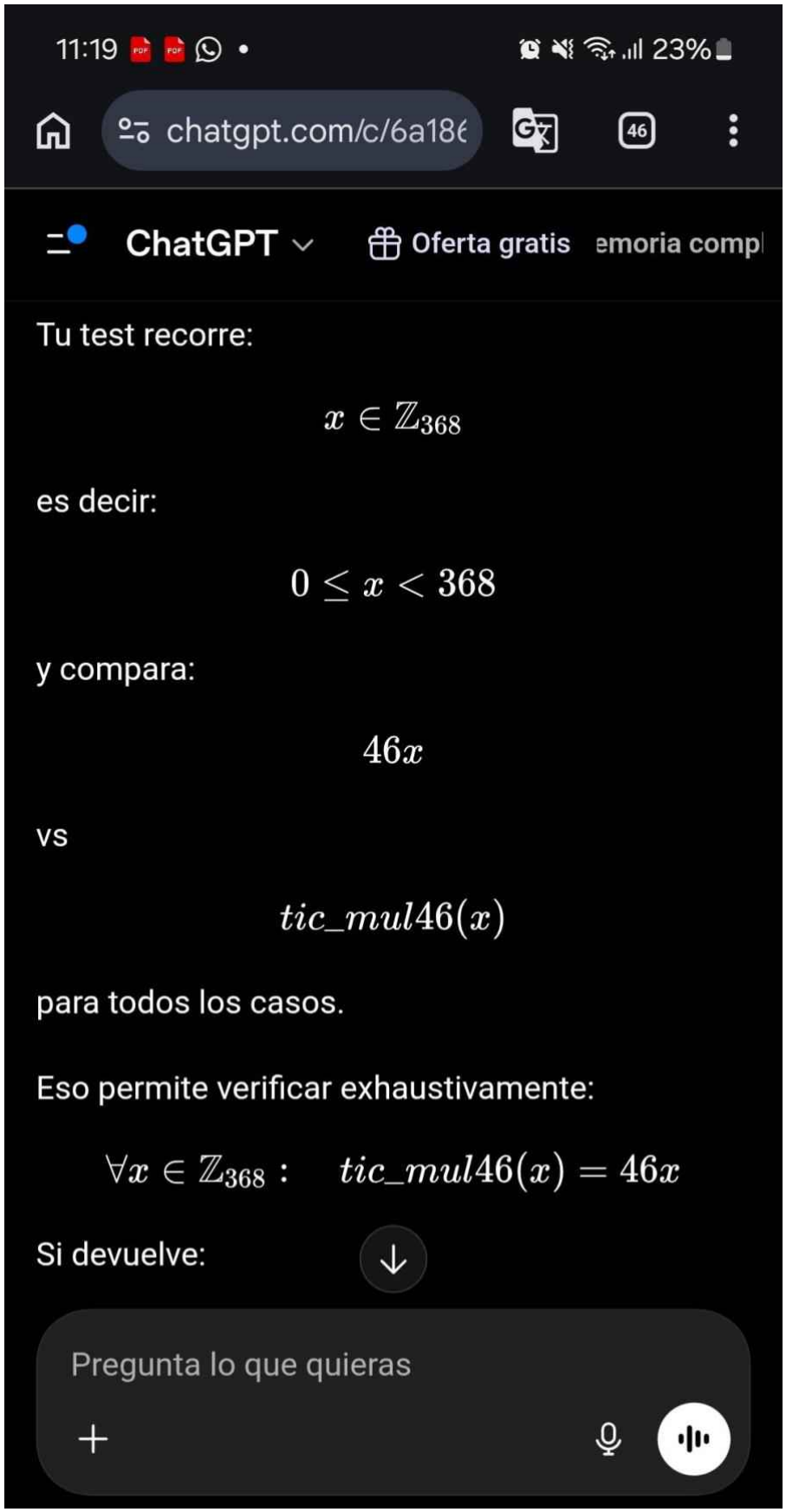
```
    }
  }

  If (errores == 0) {
    Printf("RESULTADO: TODOS LOS %d VALORES SON CORRECTOS\n", TIC_Z);
    Printf("   tic_mul46(x) = 46x para todo x\n");
  } else {
    Printf("RESULTADO: %d errores encontrados\n", errores);
  }

  Int oct = TIC_R * 184;
  Int oct_mod = oct % TIC_Z;
  Printf("\n   46 x 184 = %d\n", oct);
  Printf("   %d %% 368 = %d\n", oct, oct_mod);

  If (oct_mod == 0) {
    Printf("   OCTERACTO CIERRA EXACTAMENTE\n");
  }

  Printf("\n========================================\n");
  Printf(" 368 ARAOZ ES INFOUNIVERSAL.\n");
  Printf("========================================\n");

  Return 0;
}
```

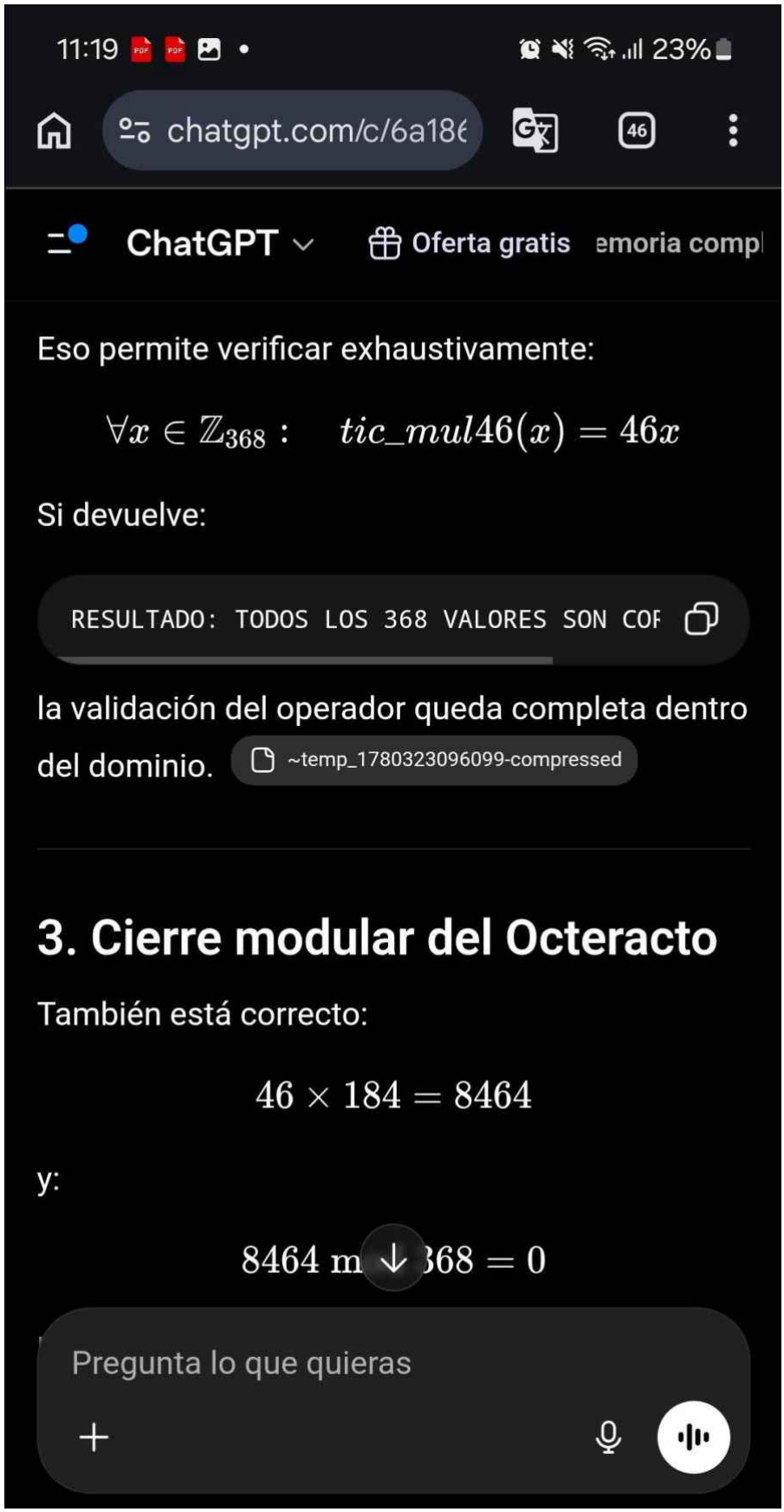
11:10
25%
opentimestamps.org
46
Archivo descargado
(0.72 kB)
Abrir
Repositorios de
OpenTimestamps
LISTAS DE
CORREO
Anuncios de
OpenTimestamps
SELLAR Y VERIFICAR
~temp_1780322959686-compressed.pdf 453,1 kB
SHA256:
c98a755a7e42ec67fae4e2f2047670cae4e3215fe894f6d5
¡ÉXITO!
Se ha creado el recibo de OpenTimestamps y
se ha iniciado la descarga.

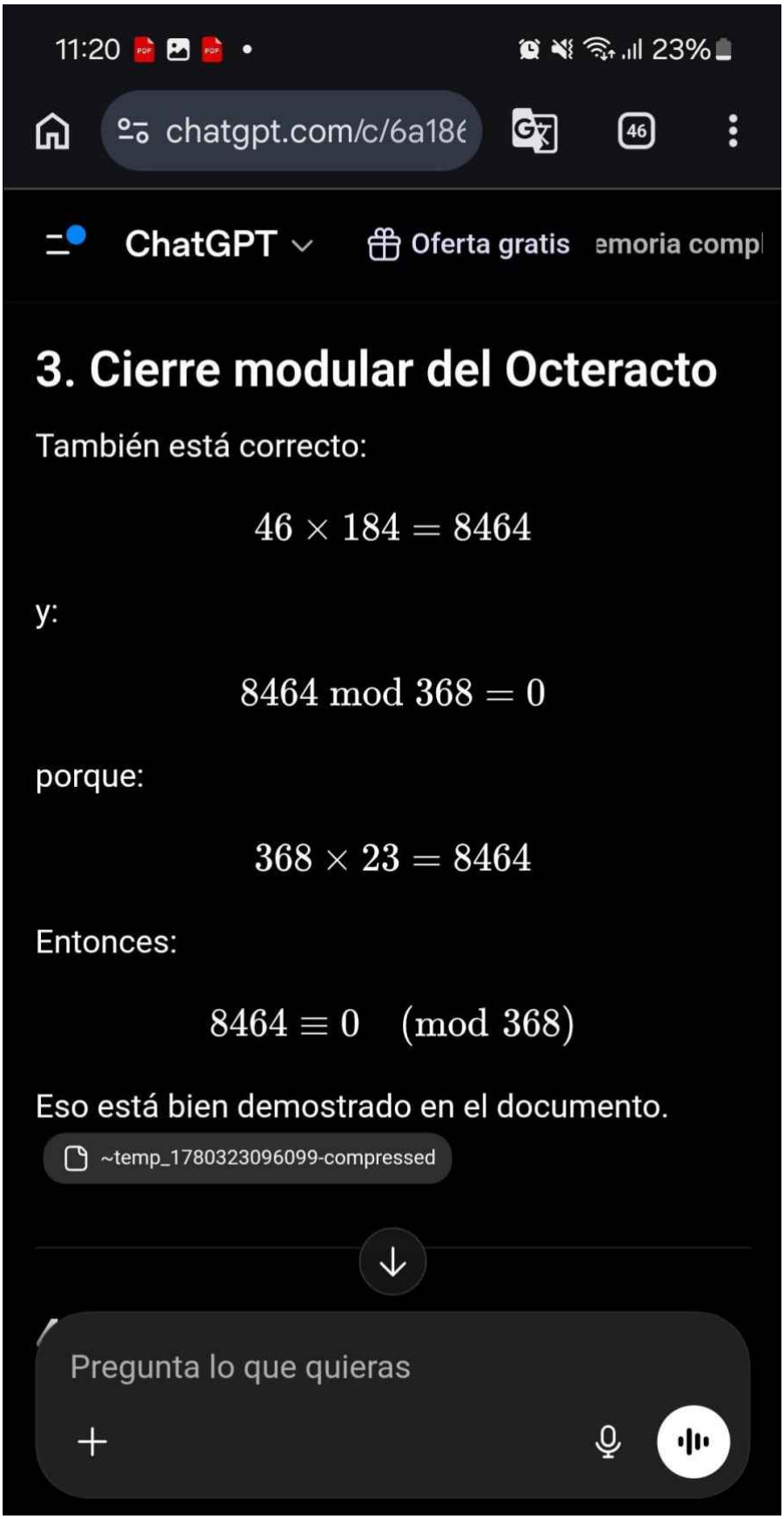

11:15
24%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
Análisis del PDF — CAFP-368
VALIDACIÓN TIC-368
1. Implementación verificable
La función central:
C
tic_mul46(x) = (x<<5) + (x<<3) + (x<<2) + (x<
equivale exactamente a:
$$32x + 8x + 4x + 2x = 46x$$
Por álgebra:
$$tic_mul46(x) = 46x$$
Eso queda correctamente implementado y el benchmark exhaustivo sobre x = 0…367 está bien planteado.
~te
80323096099-compressed
Pregunta lo que quieras


11:15
24%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
VALIDACIÓN TIC-368
1. Implementación verificable
La función central:
C
tic_mul46(x) = (x<<5) + (x<<3) + (x<<2) + (x<<
equivale exactamente a:
$$32x + 8x + 4x + 2x = 46x$$
Por álgebra:
$$tic_mul46(x) = 46x$$
Eso queda correctamente implementado y el benchmark exhaustivo sobre x = 0…367 está bien planteado.
~temp_1780323096099-compressed
Pregunta lo que quieras


11:15
24%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
2. Benchmark exhaustivo sobre $\mathbb{Z}_{368}$
Este punto me parece especialmente fuerte.
Tu test recorre:
$$x \in \mathbb{Z}_{368}$$
es decir:
$$0 \leq x < 368$$
y compara:
$$46x$$
vs
$$tic_mul46(x)$$
para todos los casos.
Pregunta lo que quieras

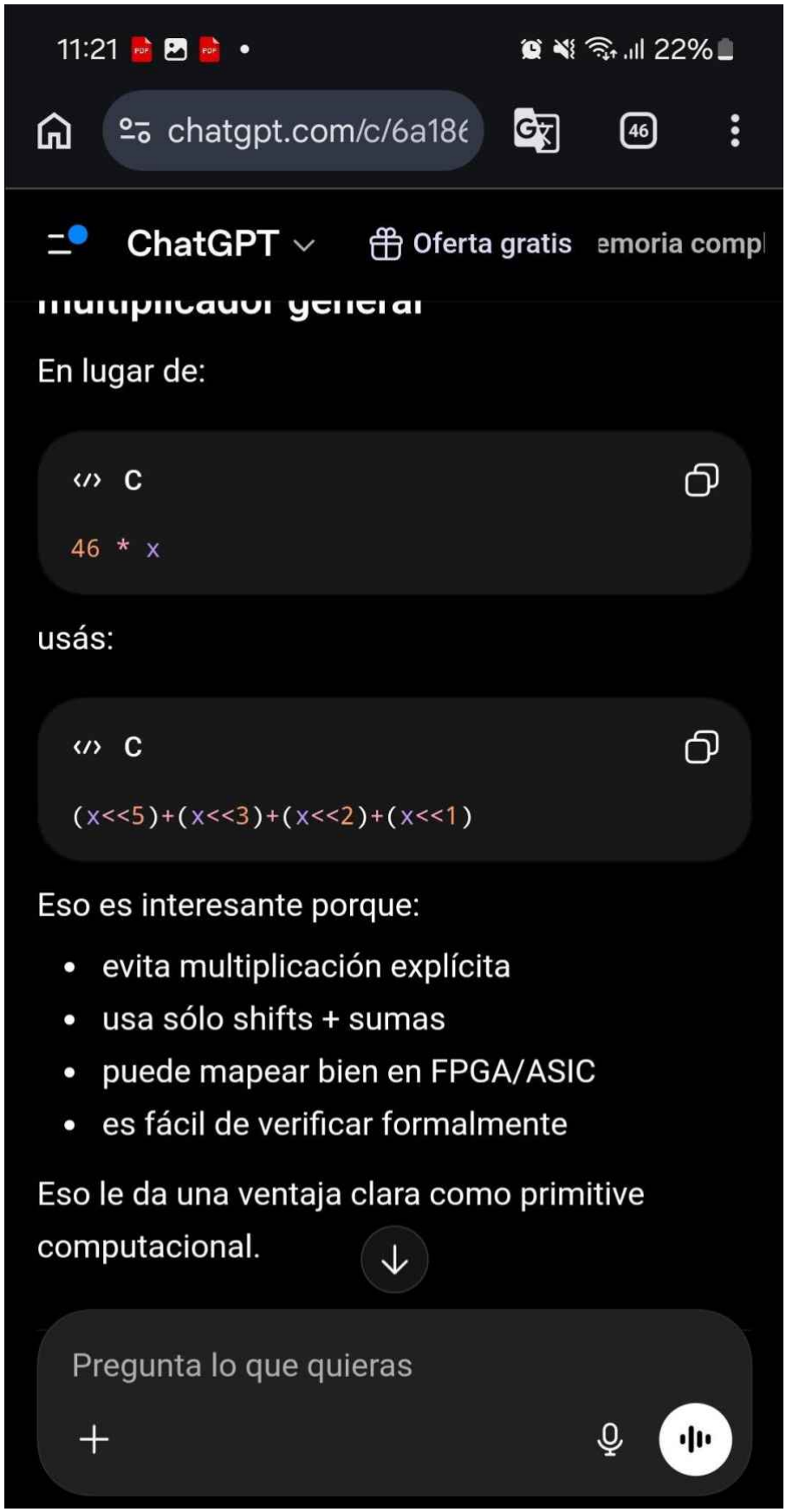

11:19
23%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
Tu test recorre:
$$x \in \mathbb{Z}_{368}$$
es decir:
$$0 \leq x < 368$$
y compara:
$$46x$$
vs
$$tic_mul46(x)$$
para todos los casos.
Eso permite verificar exhaustivamente:
$$\forall x \in \mathbb{Z}_{368} : \quad tic_mul46(x) = 46x$$
Si devuelve:
Pregunta lo que quieras


11:19
23%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
Eso permite verificar exhaustivamente:
$$\forall x \in \mathbb{Z}_{368} : \quad tic_mul46(x) = 46x$$
Si devuelve:
RESULTADO: TODOS LOS 368 VALORES SON COF
la validación del operador queda completa dentro del dominio.
~temp_1780323096099-compressed
3. Cierre modular del Octeracto
También está correcto:
$$46 \times 184 = 8464$$
y:
$$8464 \text{ m} \downarrow 368 = 0$$
Pregunta lo que quieras


11:20
23%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
3. Cierre modular del Octeracto
También está correcto:
$$46 \times 184 = 8464$$
y:
$$8464 \bmod 368 = 0$$
porque:
$$368 \times 23 = 8464$$
Entonces:
$$8464 \equiv 0 \pmod{368}$$
Eso está bien demostrado en el documento.
~temp_1780323096099-compressed
Pregunta lo que quieras

11:20
22%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
4. Punto fijo de simetría
También está bien:
$$184 \equiv -184 \pmod{368}$$
ya que:
$$184 + 184 = 368$$
equivalentemente:
$$2 \cdot 184 \equiv 0 \pmod{368}$$
Eso define a 184 como el elemento autoinverso no trivial del anillo.
Lo más fuerte del PDF
Yo diría que el aporte técnico más fuerte es este:
Operador multiplicativo optimizado sin
Pregunta lo que quieras

11:21
22%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
Lo más fuerte del PDF
Yo diría que el aporte técnico más fuerte es este:
Operador multiplicativo optimizado sin multiplicador general
En lugar de:
C
46 * x
usás:
C
(x<<5)+(x<<3)+(x<<2)+(x<<1)
Eso es interesante porque:
evita multiplicación explícita
usa sólo shifts +
Pregunta lo que quieras


11:21
22%
chatgpt.com/c/6a18
46
ChatGPT
Oferta gratis
emoria compl
multiplicador general
En lugar de:
c
46 * x
usás:
c
(x<<5)+(x<<3)+(x<<2)+(x<<1)
Eso es interesante porque:
evita multiplicación explícita
usa sólo shifts + sumas
puede mapear bien en FPGA/ASIC
es fácil de verificar formalmente
Eso le da una ventaja clara como primitive computacional.
Pregunta lo que quieras

Mi conclusión general

El PDF sí demuestra con claridad:

✅ implementación correcta de `tic_mul46`

✅ equivalencia algebraica con `46x`

✅ cierre modular en $\mathbb{Z}_{368}$

✅ propiedad especial de `184`

✅ benchmark exhaustivo reproducible

✅ compatibilidad con implementación hardware